Dendritic Cell Algorithm Enhancement Using Fuzzy Inference System for Network Intrusion Detection

Noe Elisa, Longzhi Yang
Department of Computer and Information Sciences
Northumbria University
Newcastle upon Tyne, UK
Email:{noe.nnko, longzhi.yang}@northumbria.ac.uk

Xin Fu
School of Management
Xiamen University
Xiamen, China
Email: xfu@xmu.edu.cn

Nitin Naik
Defence School of
Communications of Information Systems
Ministry of Defense, UK
Email: nitin.naik100@mod.gov.uk

***Abstract*—Dendritic cell algorithm (DCA) is an immune-inspired classification algorithm which is developed for the purpose of anomaly detection in computer networks. The DCA uses a weighted function in its context detection phase to process three categories of input signals including safe, danger and pathogenic associated molecular pattern to three output context values termed as co-stimulatory, mature and semi-mature, which are then used to perform classification. The weighted function used by the DCA requires either manually pre-defined weights usually provided by the immunologists, or empirically derived weights from the training dataset. Neither of these is sufficiently flexible to work with different datasets to produce optimum classification result. To address such limitation, this work proposes an approach for computing the three output context values of the DCA by employing the recently proposed TSK+ fuzzy inference system, such that the weights are always optimal for the provided data set regarding a specific application. The proposed approach was validated and evaluated by applying it to the two popular datasets KDD99 and UNSW_NB15. The results from the experiments demonstrate that, the proposed approach outperforms the conventional DCA in terms of classification accuracy.**

***Index Terms*—Fuzzy inference systems, TSK+, Dendritic cell algorithm, Danger theory, Network intrusion detection.**

I. Introduction

The ubiquitous usage of computer networks and applications has led to the proliferation of cyber-attacks trying to access, destroy, steal or corrupt the information stored in computer and networked systems. Network intrusion detection systems (NIDSs) are developed purposely to identify intruders who attempt to access network resources without authority or permission. Generally, NIDSs can be divided into Misuse-based (MNIDS) and Anomaly-based (ANIDS) [1]. In MNIDS, the signature of the known attacks are captured and stored in a database and the attacks are detected by matching their behaviours with the stored signatures [1]. Contrarily, ANIDS defines the behaviours of normal activities, then any traffic's behaviors which deviate from the pre-defined behaviours are treated as anomalies [1]. Different machine learning and artificial intelligence techniques have been exploited to develop ANIDS [2]–[5]. Artificial immune system (AIS) is a branch of computational intelligence which was introduced primarily for the purpose of developing NIDSs [6].

This work has been supported by the Commonwealth Scholarship Commission (CSC-TZCS-2017-717), and the Industry Academia Partnership Programme by Royal Academy of Engineering (IAPP1\100077).

The AIS employs mathematical and computational techniques to model the immune system behavior as a metaphor to a NIDS. The DCA is one implementation of the AIS algorithms which is developed by mimicking the immune danger theory and the functioning of human dendritic cells (DCs) [7]. To perform classification, the DCA goes through four phases namely initialization, context detection, context assessment and classification. Firstly, the DCA initialises a population of artificial DCs responsible for sampling the signals and data items (often referred to as antigens). Secondly, in its context detection phase, the DCA uses a weighted function to compute three output context values of co-stimulatory (CSM), mature DC (mDC) and semi-mature DC ($smDC$) from its three categories of input signals namely safe signal (SS), danger signal (DS) and pathogenic associated molecular pattern ($PAMP$). Subsequently, the three output context values are passed to the context assessment and classification phases to generate the final class labels.

The DCA weighted function requires users to either manually pre-define the weights, often using the original weights proposed by the immunologists, or empirically derive the weights from the training dataset [7]. Note that, such weights may not produce optimal classification results based on two reasons. Firstly, the original weights proposed by the immunologist indeed works well with the UCI breast cancer dataset, but not necessarily for others [3], [8]. Secondly, the relationship between the context outputs and signal inputs may not be linear intrinsically for a given dataset, and thus the weighted function may fail regardless how good the weights are. Therefore, the main goal of this study is to use fuzzy inference system to generate the three output context values of the DCA to address both issues.

This paper proposes a non-linear approach that employs the TSK+ fuzzy inference method to compute the three output context values of the DCA [9]. Note that the network datasets are often very sparse, and TSK+ works well with sparse and imbalanced datasets; therefore, TSK+ is particularly applied in this work. In order to implement this approach, a data-driven rule base generation method is employed to generate three TSK fuzzy rule bases corresponding to the three output contexts of CSM, $smDC$ and mDC. Then, given the three input signal values, TSK+ fuzzy inference method is utilised

to generate the values of CSM, $smDC$ and mDC from the corresponding TSK fuzzy rule base. Finally, the three context values are correspondingly accumulated in each DC and passed to the context assessment and classification phases to generate the final classification result. The proposed approach is validated and evaluated using two datasets, including the KDD99 [10] and the UNSW_NB15 [11]. The experimental results demonstrate the effectiveness of the proposed approach in improving the performance of the conventional DCA.

The remainder of this paper is structured as follows. Section II briefly provides the background on the TSK+ fuzzy inference system and DCA. Section III details the proposed approach of DCA context detection using TSK+. Section IV demonstrates the experimentation process followed by the validation and evaluation of the proposed approach and discussions. Section V concludes this study and points out the possible future directions.

II. Background

This section provides the background information on the TSK+ fuzzy inference system and the DCA algorithm.

A. TSK+ Fuzzy Inference System

Fuzzy inference systems use fuzzy set theory to map a given input feature space to the output space. They have been used in different computer domains to process input data to the intended output values [2], [5], [12]. They are generally represented by two types of models, Mamdani-type [13] and Takagi-Sugeno-Kang (TSK)-type [14]. The Mamdani-type is able to deal with human natural language, thus, requires defuzzification of a fuzzy output. In contrast, the TSK-type produces a crisp value which makes it more useful when crisp outputs are required. The two conventional models cannot work with highly imbalanced or sparse input observations since they require a dense rule base for which the entire input domain is fully covered. Originally introduced by [12], fuzzy interpolation extends the two models to obtain certain conclusion when a given input observation does not overlap with any rule antecedents in the rule base. A number of fuzzy interpolation techniques have been introduced in the literature [15]–[18].

Traditional TSK fuzzy inference systems are only applicable to problems with dense rule bases [9], but TSK+ allows inference to be made with sparse, dense, or imbalanced rule bases. Using TSK+, consequence of an uncovered input can be interpolated from its neighboring rules [9]. Fuzzy interpolation also simplifies fuzzy models by omitting those rules which can be approximated from their neighbouring ones in the rule base. Consider a sparse TSK rule base which contains the following n rules:

$$\begin{aligned} R_1 &: \textbf{IF } x_1 \text{ is } Y_1^1 \text{ and } \cdots x_j \text{ is } Y_j^1 \cdots \text{ and } x_m \text{ is } Y_m^1 \\ &\quad \textbf{THEN } w = f_1(x_1, \cdots, x_m), \\ &\quad \ldots \ \ldots \\ R_n &: \textbf{IF } x_1 \text{ is } Y_1^n \text{ and } \cdots x_j \text{ is } Y_j^n \ \cdots \text{ and } x_m \text{ is } Y_m^n \\ &\quad \textbf{THEN } w = f_n(x_1, \cdots, x_m), \end{aligned} \tag{1}$$

where $Y_j^i, (i \in \{1, 2, \cdots, n\}$ and $j \in \{1, 2, \cdots, m\})$, represents a normal and fuzzy convex polygon that can be conveniently denoted as $(y_{j1}^i, y_{j2}^i, \cdots, y_{jv}^i)$, v indicates the number of odd points of the fuzzy set. Consider an input observation $I = (Y_1^*, Y_2^*, \cdots, Y_m^*)$, TSK+ uses the following steps to generate a crisp inference result:

Step 1: Calculate the similarity degrees between the given input observations $(Y_1^*, Y_2^*, \cdots, Y_m^*)$ and every individual rule R_i in the rule base with the rule antecedents $(Y_1^i,\ Y_2^i,\ Y_3^i, \cdots, Y_m^i)$ using the following equation:

$$S(Y_j^i, Y_j^*) = \left(1 - \frac{\sum_{q=1}^{v} |y_{jq}^i - y_{jq}^*|}{v}\right) \cdot (Df) \ , \tag{2}$$

where Df is known as $distance\ factor$, which is used to determine the distance between the two fuzzy sets. The value of Df is determined by:

$$Df = 1 - \frac{1}{1 + e^{(-cd+5)}} \ , \tag{3}$$

where c is an adjustable sensitivity factor, and d represents the distance (usually Euclidean distance) between the two fuzzy sets. Particularly, c is a positive real number. Note that, the smaller value of c leads to a similarity degree which is more sensitive to the distance between the two fuzzy sets.

Step 2: Compute the firing strength of each rule by aggregating the similarity degrees between the given input observation and its antecedents using Equation 4:

$$\alpha_i = S(Y_1^*, Y_1^i) \wedge S(Y_2^*, Y_2^i) \wedge \cdots \wedge S(Y_m^*, Y_m^i) \ , \tag{4}$$

where $\wedge$ is a t-norm operator often implemented as a minimum operator.

Step 3: The final output is generated by aggregating the sub-consequences from all rules using Equation 5:

$$w = \sum_{i=1}^{n} \alpha_i \cdot f_n(x_1, \cdots, x_m) \ / \ \sum_{i=1}^{n} \alpha_i \ . \tag{5}$$

B. Dendritic Cell Algorithm

The DCA is used to classify input data instances either as normal (i.e., semi-mature context) or as anomalous (i.e., mature context) [7]. The key steps of the DCA is reviewed in this section.

1) Dataset Pre-processing: Feature selection approach is firstly applied to select the most important features, and there are many feature selection approaches available in the literature, such as hierarchical quotient spaces-based feature selection [19]. The information gain method is often used with the DCA algorithm due to its efficiency and effectiveness. Typically, the features with high information gain are ranked higher than other because they have stronger influence in classifying the data instances [20].

Briefly, given a dataset S, the information gain of an attribute F can be evaluated by using Equation 6 [20]:

$$G(S,F) = H(S) - \sum_{v \in values(F)} \frac{|S_v|}{|S|} * H(S_v), \quad (6)$$

where $values(F)$ represents the whole set of potential values that attribute F may take, S_v is a subset of S each having value v for attribute F, G is the information gain, and H is the entropy. In particular, the entropy H is computed as:

$$H(S) = \sum_{i=1}^{i=2} -p_i * log_2 p_i, \quad (7)$$

where p_i is the probability of class i in the dataset S based on the values of attribute F. The higher the entropy is, the higher the information the corresponding attribute provides. A threshold is set so that, only attributes with higher information gains than this threshold are retained in the dataset for further processing.

2) Signal Categorisation: After data pre-processing, the DCA categorises the input features into three signal groups of SS, DS and $PAMP$. In order to find a relevant subset of features for each signal category, in this work, signal categorisation is performed by maximising the feature-class mutual information. The mutual information, $I(F;C)$ between two random features F and C is the amount of information that C gives about F. $I(F;C)$ is calculated using the following equation:

$$I(F;C) = \sum_{f \in values(F), c \in values(C)} p(f,c) * log(\frac{p(f,c)}{p(f)p(c)}), \quad (8)$$

where $p(f,c)$ is the joint probability of values of f and c being taken, $p(f)$ and $p(c)$ are the marginal probability of attributes values f and c being taken respectively.

The following two steps are used to categorise each selected feature in Section II-B1 to its signal category.

Step 1: Compute the feature-class mutual information between each selected attribute and each class presented in the dataset (i.e., normal class and anomalous class) by using Equation 8.

Step 2: If an attribute has a higher mutual information with the normal class (maximising) and significant lower mutual information with the anomalous class (minimising), it is categorised as SS. If an attribute has higher mutual information with the anomalous class (maximising) but significant lower mutual information with the normal class (minimising), it is categorised as PAMP. Otherwise, the feature is categorised as DS.

3) Data Sampling through DCs: Subsequently, a population of artificial DCs is initialised in a pool and the DCA moves to the sampling stage. Generally, for most of the implementation, the population contains 100 DCs [7]. Each DC randomly samples data items. Also, each DC is assigned a migration threshold to limit the time-span it spends for data items sampling; a DC matures once it stops sampling data based on a pre-defined threshold. A pre-defined number of mature DCs (often 10) are selected randomly from the pool for further processing in three DCA phases below [7]. Note that, once the DCs are done with the classification of data items they have sampled, DCs are reset and returned to the sampling population in order to maintain the population size. This process continues for the implementation of an NIDS along with the collection of new input data in time line.

1. DC Context Detection: In this phase, the selected DCs for sampling use a set of pre-defined weights and the signal values to compute CSM, $smDC$ and mDC by employing a weighted summation function as expressed as follows:

$$C[CSM, smDC, mDC] = \frac{(W_{PAMP} * C_{PAMP}) + (W_{SS} * C_{SS}) + (W_{DS} * C_{DS})}{W_{PAMP} + W_{SS} + W_{DS}}, \quad (9)$$

where C_{PAMP}, C_{DS} and W_{SS} are the values of $PAMP$, DS and SS respectively, which are generated by aggregating the assigned features values. The weights (W_{PAMP}, W_{SS} and W_{DS}) are pre-defined by the user or derived empirically from the dataset. Note that, DCs accumulate the values of CSM, $smDC$ and mDC of all the data items they sample overtime. As soon as the cumulative CSM exceeds the assigned migration threshold, the DCs cease to sample any more data items and then move to the context assessment phase. In this process, a single data instance may be sampled by multiple DCs. The CSM's migration threshold is determined from the characteristic behaviour of the dataset and the amount of data instances the DCs can collect.

2. DC Context Assessment: The values of the cumulative $smDC$ and mDC obtained from the context detection phase are compared in each DC. If the value of the $smDC$ is greater than mDC, then a DC goes to semi-mature context (i.e., context=0, and in this case each DC regards the corresponding data items as normal), otherwise it goes to mature context (i.e., context=1, and in this case, each DC regards the data item as anomalous).

3. Classification: The mature context antigen value ($MCAV$) of each collected data item is computed in this phase, by dividing the number a data item being presented in the mature context (anomalous) to the total number of times it is presented by DCs. If the $MCAV$ value is greater than the $MCAV$'s anomaly threshold, the data item is classified as anomalous otherwise normal. The $MCAV$'s anomaly threshold is often determined from the training dataset by taking the percentage of the anomalous records present.

III. DC Context Detection using TSK+

This work proposes a new approach for DC context detection using TSK+, as shown in Figure 1. Without losing generality, suppose a DC sampled m data items (i.e., $di_1, di_2 \ldots di_m$,), and stops sampling as its accumulated CSM value reaches the threshold. The DCA algorithm constantly calculates the CSM, $smDC$ and mDC value for each

data item di_j $(1 \leq j \leq m)$ using the TSK+ inference approach. Then, the DC's cumulative CSM, $smDC$ and mDC values are calculated by accumulating the CSM, $smDC$ and mDC values for all the data items in the DC for the next phase DC context assessment.

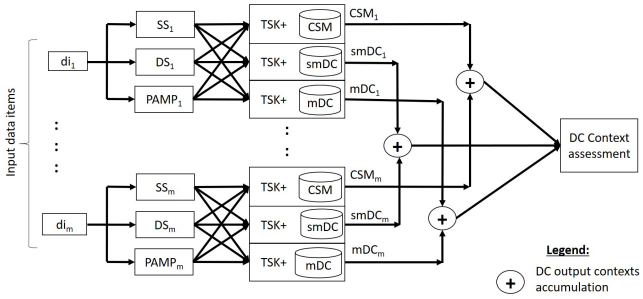

Figure 1: Overview of the proposed system

A. Overview

For a given data item with its SS, DS, and $PAMP$ values, the TSK+ approach reviewed in Section II-A takes these values as input and produces the CSM, $smDC$ and mDC for this data instance as output. A TSK+ fuzzy inference approach is comprised of two parts, an inference engine and a rule base. The inference engine has been presented in Section II-A, therefore, the focus here is the generation of the fuzzy rule bases.

There are generally two ways for rule base generation, data-based or knowledge-based. This work takes the data-based approach. Take the CSM rule base generation as an example in this subsection. The rule base is initialised by using an artificial training dataset calculated using Equation 9 for a given training data set; and each data instance has four features $(SS, DS, PAMP, CSM)$. The process of the rule base generation for CSM is illustrated in Figure 2, which consists of three main steps, including clustering, rule base initialisation and optimisation, as detailed in the following subsections.

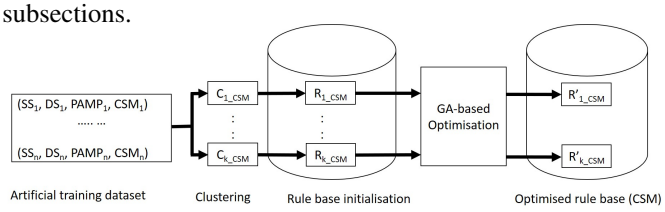

Figure 2: The TSK+ fuzzy rule base generation

B. Clustering

The K-Means clustering algorithm is applied to the artificial data set first to group similar artificial data instances into clusters based on the three input features (i.e., SS, DS, and $PAMP$) and one output feature (i.e., CSM). Note that the number of clusters needs to be determined for the clustering algorithm, which can be implemented in multiple ways. The Elbow method is particularly utilised in this work to predetermine the number of clusters k for the K-Means algorithm due to is popularity [9].

C. Rule Base Initialisation

Once the required clusters have been formed, each of the k individual cluster is used to define one TSK fuzzy rule, which jointly initialise the raw rule base. Given the ith determined cluster c_i consisting of items represented by their three input signals associated with the output CSM context, the corresponding TSK fuzzy rule R_i can be generated as follows:

$$\begin{aligned} R_i : \textbf{IF } SS \text{ is } Y_1^i \text{ and } DS \text{ is } Y_2^i \text{ and } PAMP \text{ is } Y_3^i, \\ \textbf{THEN } CSM = f_i(SS, DS, PAMP), \end{aligned} \tag{10}$$

where Y_v^i $(v = (1, 2, 3))$ are fuzzy sets representing the cluster on its SS, DS, and $PAMP$ dimensions, respectively. For simplicity, this work utilizes triangular membership functions, therefore $Y_v^i = (y_{v1}^i, y_{v2}^i, y_{v3}^i)$. The core of the fuzzy set is defined as the cluster centre, whilst the support of the fuzzy set is obtained from the span of the cluster.

The consequent of the CSM TSK fuzzy rule regarding cluster c_i is a polynomial function of the SS, DS and $PAMP$ values for any given input observation. In this work, a first order polynomial is used due to its simplicity, which is given by:

$$\begin{aligned} CSM = f_i(SS, DS, PAMP) = \rho_0^i + \rho_1^i SS + \rho_2^i DS \\ + \rho_3^i PAMP, \end{aligned} \tag{11}$$

where $\rho_0^i, \rho_1^i, \rho_2^i, \rho_3^i$ are constant parameters.

D. Rule Base Optimisation

The genetic algorithm (GA) is employed in this work to optimise the parameters of the raw rule base in this work. The GA is chosen in this work due to its ability to easily achieve the better balance between exploitation and exploration of search space simply by setting well its parameters (i.e.; mutation and crossover rates). The GA utilises adaptive metaheuristic search techniques to find an approximate solution to the optimization problems. The algorithm starts by initialising a population with random individuals. It then uses the techniques inspired by evolutionary biology such as mutation, selection, crossover and elitism to evaluate each individual in every generation and selects the best individuals who survive to the next generation. Gradually, more effective individuals are evolved over several iterations until a specified level of performance or maximum number of generations is reached.

In this work, the initial population $\mathbb{P} = \{I_1, I_2, ..., I_{|\mathbb{T}|}\}$ is formed by taking the initialised raw rule base and its random variations. Here, I represents an individual and $|\mathbb{T}|$ is the size of the population which is a problem-specific adjustable parameter, typically in a range from tens to thousands, with 20-50 being widely used [5], [9].

Assume that an initialised CSM raw rule base contains k rules as shown in Figure 2. In each population $(\mathbb{P})$, an individual (I) is designated as a possible solution that comprises of all parameters of the polynomial functions for the TSK rule consequent, which is given by $I = \{\rho_0^1, \rho_1^1, \rho_2^1, \rho_3^1, ..., \rho_0^i, \rho_1^i, \rho_2^i, \rho_3^i, ..., \rho_0^k, \rho_1^k, \rho_2^k, \rho_3^k\}$, where k is

the total number of rules in the current raw rule base. Note that the individuals may be designed to also include the parameters for all fuzzy sets, but this is not considered in this work for efficiency. In this work, a single point crossover and mutation are applied.

Note that, the rule base generation process for CSM is also applied for $smDC$ and mDC, resulting in three rule bases to support the DCA. The performance of each individual is evaluated by the classification accuracy of the DCA. More concretely, the values of CSM, $smDC$ and mDC computed from the resultant polynomial functions are used by the DCA algorithm to determine the classification accuracy for each individual. In this system, GA terminates when the pre-specified maximum number of iterations is reached or the classification accuracy of the DCA exceeds the pre-defined threshold of the optimum accuracy. When the GA terminates, the optimal solution is taken from the fittest individual in the current population. From this, the optimised constant parameters are used to form the optimised rule base for the CSM output context, so as for $smDC$ and mDC.

Once the three optimised rules bases for CSM, $smDC$ and mDC are generated, they can be used to work with the DCA for classification tasks by providing the context values for each DC. From this, the DCA algorithm continues for the context assessment stage and the classification stage as introduced in Section II-B.

IV. Experimentation

The proposed approach is validated using 10% of the KDD99, and UNSW_NB15 datasets. All attack types present in these datasets are categorised into anomaly class.

A. KDD99 Dataset Description

The KDD99 dataset was generated to support the development of NIDSs [10]. The dataset is divided into two sets for training and testing. Each data item has 41 features. The training dataset contains 494,021 records (97,278 normal and 396,743 anomalous) while the testing dataset comprises of 311,029 records (60,593 normal and 250,436 anomalous).

B. UNSW_NB15 Dataset Description

The UNSW_NB15 dataset was made publicly available in 2015 to support the evaluation of modern NIDSs [11]. It contains nine new modern attack types which are not present in the KDD99 dataset including Reconnaissance, Shellcode, Exploit, Fuzzers, Worm, DoS, Backdoor, Analysis and Generic. It is further divided into training and testing sets. Each data instance in this data set has 49 features including the class label. The training dataset contains 175,341 records (56,000 normal and 119,341 anomalous) while the testing dataset comprises of 82,332 records (37,000 normal and 45,332 anomalous).

C. Dataset Pre-processing

Once the information gain method is applied to the two training datasets, ten and seventeen features are selected for the KDD99 and UNSW_NB15 datasets, respectively. Consequently, the feature-class mutual information is used to categorise the features into SS, DS and $PAMP$. Each feature is normalized using the min-max normalisation as in [5], then the value of each signal is equal to the average ot the assigned features. The features were categorised as follows.

For the KDD99 dataset, DS={count and srv_count}, SS={logged_in, srv_different_host_rate and dst_host_count}, $PAMP$={serror_rate, srv_serror_rate, same_srv_rate, dst_host_serror and dst_host_rerror_rate}.

For the UNSW_NB15 dataset, DS={sbytes, dbytes, dload and dmean}, SS={dpkts, sttl, smean, ct_state_ttl, ct_dst_sport_ltm and ct_srv_dst }, $PAMP$={dur, rate, dttl, sload, ct_srv_src, ct_src_dport_ltm and ct_dst_src_ltm}.

D. DCs Initialisation and Sampling

A population of 100 DCs is used and 10 DCs are selected at once to sample the signals and data items as it was a common practice in [3], [7]. From the characteristic behavior of the two datasets, the CSM's migration threshold is set to 300 and 250 for the KDD99 and UNSW_NB15 dataset respectively.

E. TSK Rule Base Generation

The TSK fuzzy rule base is generated using the following steps:

Step 1: The Optimal Number of Clusters for Each Context: The optimal number of clusters generated after applying the Elbow method and K-Means clustering method for each output context are highlighted in Table I.

Table I: The number of clusters for each context

	CSM	$smDC$	mDC
Number of clusters	7	7	7

Step 2: TSK Fuzzy Rule Extraction: Each determined cluster is used to form a rule antecedent contributing to a total of seven (7) rule antecedents in each rule base. The GA parameters used to optimise the constant parameters of the polynomial functions of the TSK consequent are; mutation rate of 0.1, crossover rate of 0.95, 250 number of iterations and 50 number of individuals in a population.

Details are omitted to conserve space, but the TSK+ examples from fuzzy rule generation to classification using few data items are demonstrated step by step in [5], [9].

F. DCA Context Assessment and Classification

The percentage of anomalous data items present in the KDD99 training dataset is 0.80 (i.e., 396,743 out of 494,021 records), therefore, the $MCAV$'s anomaly threshold is set to 0.80. Similarly, the percentage of anomalous data items present in the UNSW_NB15 training dataset is 0.68 (i.e., 119,341 out of 175,341 records), therefore, the anomaly threshold is set to 0.68.

G. Results and Discussion

The classification performance obtained for the KDD99 and UNSW_NB15 training datasets are 97.89% and 92.45% respectively whilst the testing datasets for the KDD99 and UNSW_NB15 produced a classification performance of

92.78% and 89.30% respectively as shown in Table II. The proposed approach is also compared with the conventional DCA with its weighted function optimised using the technique proposed by [3].

Table II: Classification Results for the KDD99 and UNSW_NB15 Dataset

Dataset	TSK+ approach		Conventional DCA	
	Accuracy (%)		Accuracy (%)	
	Training	Testing	Training	Testing
KDD99	**97.89**	**92.78**	97.29	85.34
UNSW_NB15	**92.45**	**89.30**	87.65	78.04

The comparison results presented in Table II indicate that, the proposed approach has higher classification accuracy for both training and testing datasets. Also, the difference between the training and testing accuracies of the proposed approach are 5.44% and 3.48% for the KDD99 and UNSW_NB15 dataset respectively while for the conventional DCA are 7.99% and 7.30% for the KDD99 and UNSW_NB15 dataset respectively. Compared to the conventional DCA, the proposed approach has smaller variation between the training and testing classification accuracy. This shows that, the TSK+ system has better generalisation compared to the weighted function of the DCA. Therefore, TSK+ method is more useful in calculating the output context values of the DCA.

Additionally, TSK+ system helps to eliminate the errors that user may make when defining or generating the weights for the weighted function of the DCA. The fine-tuning process of the testing accuracies for the two datasets over 250 iterations is captured in Figure 3. The testing time required by the proposed approach in each single iteration is the same as that of the conventional DCA, the differences lie on the training time where the proposed approach requires more time.

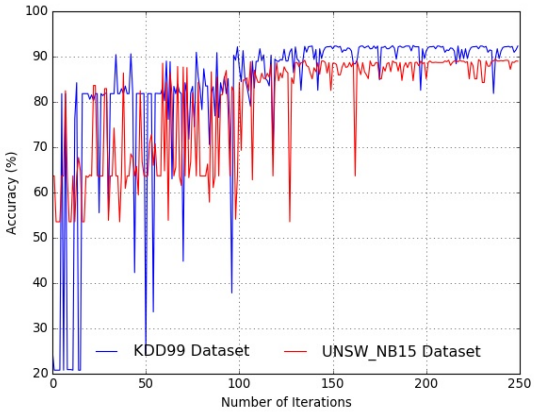

Figure 3: Fine-tuning the testing accuracies

V. Conclusions

This work proposed an enhanced DCA approach using the TSK+ system. In particular, the proposed approach employs a data-driven fuzzy rule base generation method to extract the TSK fuzzy rules from the three input signals of the DCA and generate three TSK fuzzy rule bases corresponding to CSM, $smDC$ and mDC output contexts. The experimental results using the KDD99 and UNSW_NB15 datasets demonstrate that, compared to the conventional DCA, the TSK+ method is more effective in computing the output context values. The possible future work is to use this approach to adaptively derive and fine-tune the parameters of the rule base using online real time traffics. Furthermore, TSK+ method can be extended to the context assessment phase of the DCA to smooth the difference between cumulative mDC and $smDC$ context for aiming for classification result.

References

[1] Giovanni Vigna and Richard A Kemmerer. Netstat: A network-based intrusion detection system. *Journal of computer security*, 7(1):37–71, 1999.

[2] Longzhi Yang, Jie Li, Gerhard Fehringer, Phoebe Barraclough, Graham Sexton, and Yi Cao. Intrusion detection system by fuzzy interpolation. In *Fuzzy Systems (FUZZ-IEEE), 2017 IEEE International Conference on*, pages 1–6. IEEE, 2017.

[3] Noe Elisa, Longzhi Yang, and Nitin Naik. Dendritic cell algorithm with optimised parameters using genetic algorithm. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2018.

[4] Noe Nnko, Longzhi Yang, Yanpeng Qu, and Fei Chao. A revised dendritic cell algorithm using k-means clustering. pages 1–8, 2018.

[5] Noe Elisa, Jie Li, Zheming Zuo, and Longzhi Yang. Dendritic cell algorithm with fuzzy inference system for input signal generation. In *UK workshop on computational intelligence*, pages 203–214. Springer, 2018.

[6] Steven A Hofmeyr and Stephanie Forrest. Architecture for an artificial immune system. *Evolutionary computation*, 8(4):443–473, 2000.

[7] Julie Greensmith, Uwe Aickelin, and Steve Cayzer. Introducing dendritic cells as a novel immune-inspired algorithm for anomaly detection. In *International Conference on Artificial Immune Systems*, pages 153–167. Springer, 2005.

[8] Julie Greensmith, Uwe Aickelin, and Jamie Twycross. Articulation and clarification of the dendritic cell algorithm. In *International Conference on Artificial Immune Systems*, pages 404–417. Springer, 2006.

[9] Jie Li, Longzhi Yang, Yanpeng Qu, and Graham Sexton. An extended takagi–sugeno–kang inference system (TSK+) with fuzzy interpolation and its rule base generation. *Soft Computing*, 22(10):3155–3170, 2018.

[10] KDD Cup 1999 Data. "http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html/. Accessed: 2018-12-16.

[11] Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *Military Communications and Information Systems Conference (MilCIS), 2015*, pages 1–6. IEEE, 2015.

[12] LászlóT Kóczy and Kaoru Hirota. Approximate reasoning by linear rule interpolation and general approximation. *International Journal of Approximate Reasoning*, 9(3):197–225, 1993.

[13] Ebrahim H Mamdani and Sedrak Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International journal of man-machine studies*, 7(1):1–13, 1975.

[14] Tomohiro Takagi and Michio Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE transactions on systems, man, and cybernetics*, (1):116–132, 1985.

[15] Longzhi Yang and Qiang Shen. Adaptive fuzzy interpolation. *IEEE Transactions on Fuzzy Systems*, 19(6):1107–1126, 2011.

[16] Zhiheng Huang and Qiang Shen. Fuzzy interpolation and extrapolation: A practical approach. *IEEE Transactions on Fuzzy Systems*, 16(1):13–28, 2008.

[17] Longzhi Yang and Qiang Shen. Closed form fuzzy interpolation. *Fuzzy Sets and Systems*, 225:1–22, 2013.

[18] Longzhi Yang, Fei Chao, and Qiang Shen. Generalised adaptive fuzzy rule interpolation. *IEEE Transactions on Fuzzy Systems*, 25(4):839–853, 2017.

[19] Qiangyi Zhang, Yanpeng Qu, Ansheng Deng, and Longzhi Yang. Hierarchical quotient spaces-based feature selection. In *2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)*, pages 770–775, 2018.

[20] Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.